Supplemental:
GroundUp: Rapid Sketch-Based 3D City Massing

Gizem Esra Ünlü[1] Mohamed Sayed[2] Yulia Gryaditskaya[3] Gabriel Brostow[1]

[1]University College London [2]Niantic [3]PAI and CVSSP, University of Surrey

S1 Synthetic data

S1.1 Data generation

To train our networks, we use the UrbanScene3D dataset [4] which contains large-scale 3D models of six real-world cities. We selected New York and Chicago for training and validation respectively, and San Francisco for testing. Our training set contains $40K$ samples, our validation set contains $2K$ samples, and our test set comprises $1K$ samples. For training samples, we perform random augmentation on the heights of individual buildings by scaling each building along the vertical axis to increase the diversity of our scenes. We generate synthetic sketches of buildings in perspective views and their ground-truth depth and segmentation maps, using Blender Freestyle [1].

S1.2 View Selection in 3D cities

As we mentioned in Sec. S1.1 of the main paper, we generate synthetic sketches of buildings in perspective views and their ground-truth depth and segmentation maps, using Blender Freestyle [1]. We place two cameras for each scene: one top-down orthographic Cam_t and one aerial perspective Cam_p. We start by sampling Cam_p's location in the scene and consequently set Cam_t in the positive look-at direction of the former at the midpoint between near and far planes. To avoid placing Cam_p within building geometry, we pre-process each city and label traversable regions on the ground plane. Moreover, each camera sits at a pre-determined height above the ground, selected so that most of the buildings are observed from above. This implies that since our system was trained with fixed settings for top-down S_t and perspective S_p sketches, it expects that the inputs should adhere to these rendering settings. We recognize that this limits the choice of viewpoints, and in full-featured applications, the urban designer may want to choose a different viewpoint, such as a street-level sketch, or use an axonometric projection. However, we believe that robustness to such representation changes is only a matter of training on a dataset that includes a wider range of rendering settings.

Fig. S1: Ground-truth maps from our synthetic dataset: perspective synthetic sketches S_p, foreground masks for perspective views with visualized segmentation of buildings (in the method we only use the binary foreground mask), depth in perspective views D_p, top-down synthetic sketches S_t, building-level segmentation in top-down views $M_t^\star$ and top-down depth maps D_t. Please see Sec. S1.1 of the main paper and Sec. S1.2 for details on data generation and view selection.

S1.3 Representative samples

In Fig. S1, we provide samples from our training dataset, showing: perspective synthetic sketches S_p, foreground masks for perspective views M_p, depth in perspective views D_p, top-down synthetic sketches S_t, building-level segmentation in top-down views $M_t^\star$ and top-down depth maps D_t.

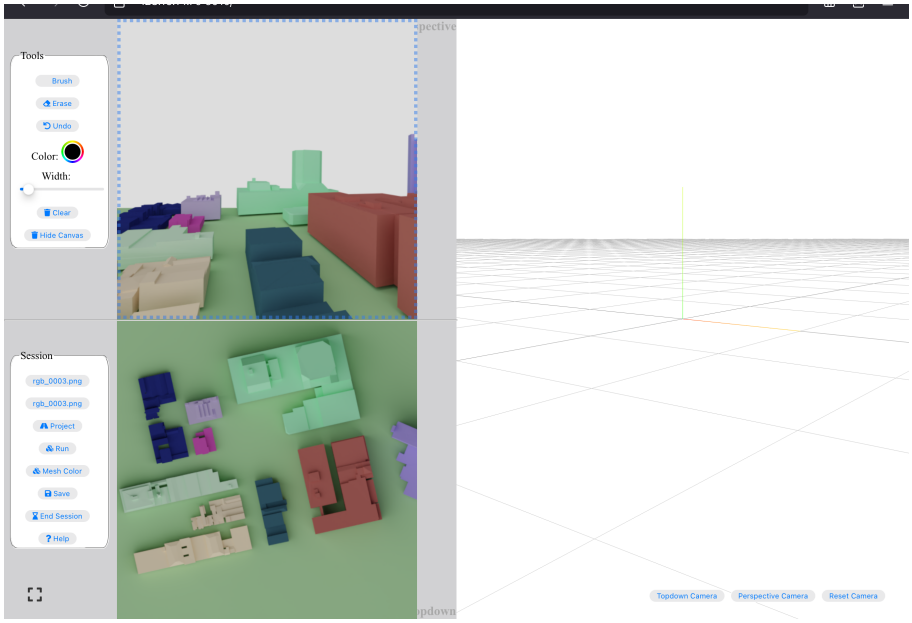


Fig. S2: Screenshot of our interface, as seen by participants in our user study. Note that both sketching views (top-down and perspective) have been loaded with reference images. This means user study participants were mostly modeling existing massing, instead of inventing new designs.

S2 User study: Modeling interface

To validate our contributions, we built an interactive user interface in HTML, JavaScript, and Python. The 3D massing system works in real-time on a Titan X, and can be used on any touch-screen device through a browser. While mouse, touch, and stylus inputs are allowed, we recommend users use a stylus, because it is easier and results in higher-quality sketches.

The interface is split into three main regions: a 3D viewport for interaction with a predicted 3D scene, and two sketching canvases for perspective and top-down views. Fig. S2 and Fig. S3 show our sketching interface, with and without a reference underlay in the sketch canvases, respectively. For sketching,

our tool supports standard capabilities such as erasing and undoing. Strokes are treated as vector data. We support two levels of zoom for the sketch views. The integrated 3D viewer is simple, and generated meshes can be exported to downstream 3D tools, *e.g.* for adding details or vectorized rendering like Fig. 1 (4) of the main paper. An important capability that was added in response to pilot users was a sketch-to-sketch projection. Users can project their top-down sketches to the perspective canvas, allowing them to see the building layouts as a kind of foundation Fig. S3. This supports users in aligning their masses between top-down and perspective sketches, which can be hard to do otherwise.

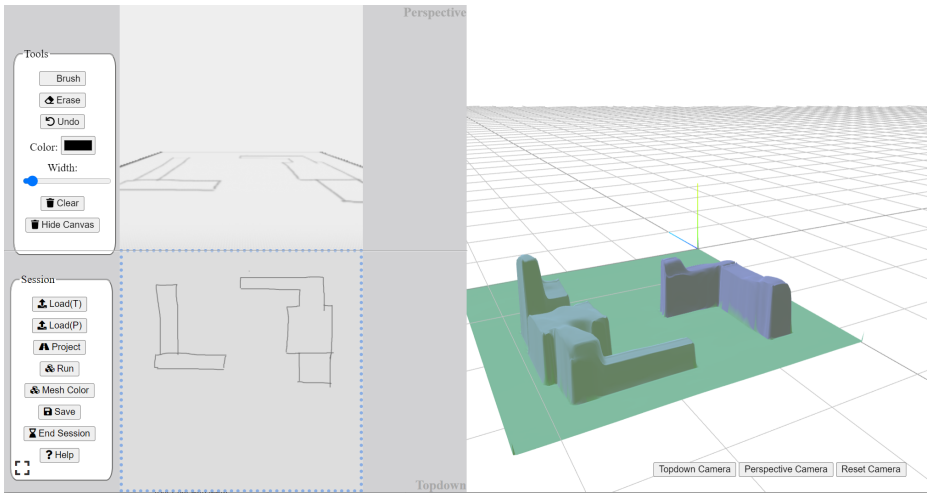


Fig. S3: Sketching interface. The interface is split into 3 major components: a 3D view for interaction with a predicted 3D scene, and two sketching canvases for perspective and top-down views. The buildings are generated here using only layout information.

S3 User study: Additional details and feedback

User feedback, especially from the post-study questionnaire, is provided in Tab. S1. Additionally, we list here quotes from all the users in our study, grouped by sentiment: good, bad, and neutral.

S3.1 Quotes with positive sentiment

- *Very cool system!*
- *Make it easy to iterate on designs. I can adapt it as I go. Deterministic behavior, so I feel that I have control over the output.*

- *I think that could be a very useful tool. Even if I sketched it really really well on paper, I'll subconsciously convince myself it works, even if in 3D it doesn't work. (e.g. gaps in their Snake-tunnel model)*
- *For massing, we always start designing from the topdown and in sketch form. I usually have an idea in mind for what the design would look like in perspective. But just from the topdown it is hard to visualize how the buildings would look like in perspective. This tool is great for visualizing the perspective view quickly.*
- *This was FAST! In Rhino I'd need at least 10 min for basic meshforming, and then 30 minutes to make that more accurate.*
- *I believe that the first step of design should begin with free hand. The software tools are limit my creativity. That's why mostly l was doing on paper sketch after that l import to revit or sketchup. I believe that style would be super useful.*
- *I feel like if you get the rough shape + layout from your sketch, then you can easily import and get full 3D, vs. just sketching on paper and then you just have a flat sketch.*
- *(good to) start to visualize geometry in the plan into 3d prespective*
- *I don't tend to design cities/buildings, so this particular implementation likely wouldn't be useful for me personally, but a more general 3D-model-from-sketch (e.g. for random objects in a room, like a couch, table, etc.), could be useful for rapidly creating AR/VR spaces.*
- *Speed of making model and quick modify is useful*
- *This is too fun!*
- *(on sketching 2 views, and the possibility of sketching more) If I had to sketch a 2nd perspective, I wouldn't think it's worth it.*
- *If I had to sketch a 2nd perspective, I wouldn't think it's worth it.*
- *Nice to use.*
- *The plan to perspective projection from perspective canvas to topdown is very useful when doing freefrom sketching.*

S3.2 Quotes with negative sentiment

- *The shape of the roof can (sic) be chosen. (likely meant can't)*
- *Only concern is how accurate it can be - I need details for only some situations*
- *It would be nice if I could edit the heights on the 3D model to what I wanted them to be (i.e. refine the 3D model by clicking and dragging on the tops of the buildings). I feel like it could also be useful to be able to quickly place trees and roads (things that aren't just buildings).*
- *Finer details are hard to sketch.*
- *Tried to draw pitched roof in the top-down: bad result.*
- *I wish I could reduce the opacity of top-down projected sketch lines in the Perspective View. They're obscuring my reference image.*
- *Just like working in my sketchbook, but you also see the 3D even if it's not perfect*

– *Depending on the design scenario, I would want to sketch from different viewpoints (for the perspective sketch). For some scenarios, a street-level sketch would be better. But for massing, a higher perspective is better. But depending on the scene I am designing, I would like to change my sketch to match the scene: I wish I could change the viewpoint for perspective sketching in this tool.*

S3.3 Quotes with neutral sentiment

– *(Please add) Zoom in, zoom out tool*
– *(Please add) Line weight to differentiate elements in sketching*
– *To write text on it, e.g. overlay window, like a post-it note on the 3D mesh. Like annotation to show where the wind goes.*
– *keyboard shortcut to switch between canvases*
– *Maybe would want an image-to-sketch converter, so I can just pull in the image and then edit lines.*
– *Would be cool to also use sketches to define building details, e.g. door and window. Could be nice to use a prompt to texture the building.*
– *I like the melty thing it created - like gipsum - I couldn't do that in Rhino really. Rhino says: "your line is this, follow it!"*
– *Details in the facade*
– *I think it would be nice to have a quick way to get a 3D representation to then get a more precise building. I could see myself tracing over with a cube [in the 3D view] - depends on the level of detail I'm going for. Normally in Blender, I'd start with a cube and position things relative to it. Could have a concrete wrapping of initial shape with a sharp convex hull. But could skip it if it's already sharp enough.*

UI just needs small refinements
– *If I want details, I'll just do it in Rhino.*

S3.4 Summary of short-answer responses to the post-study questionnaire

Tab. S1 provides extended statistics supporting the discussion in Sec. 5 of the main paper.

Post-study Question	**Architect cohort (5 respondents)**	**non-Architect cohort (5 respondents)**
Were you able to recreate the buildings in the reference image in 5 minutes?	Yes: 5/5	Yes: 2/5 No: 3/5
How accurately were you able to recreate the buildings in the reference images using the sketching interface? Scale: [-2 -1 0 +1 +2] where +2: matches reference well	+1: 4/5 0: 1/5	+2: 1/5 +1: 4/5
How likely are you to use the sketching interface in this study in the future for 3D building massing in the early design/ideation stage? Scale: [-2 -1 0 +1 +2], where -2: highly unlikely, +2: highly likely	+2: 3/5 +1: 1/5 0: 1/5	+1: 1/3 0: 2/3 (2 non-responses)
Would you consider using the sketching interface in this study as part of your 3D model creation process? For example, instead of using 3D modeling software only (e.g. Rhino), ideating in this sketching interface, before importing the output 3D mass building model into Rhino and continuing there?	Yes: 4/5 No: 1/5	Yes: 1/3 Conditional Yes: 2/3 (2 non-responses)
In the future, which one could you see yourself using for making 3D mass models?	Sketch: 3/5 Rhino: 2/5	Sketch: 3/5 Rhino: 1/5 Blender: 1/5
Would seeing a 3D model from your sketch projected to 2D help you refine your sketch? (overlaid in the 3D canvas)	Yes: 5/5	Yes: 4/5 No: 1/5

Table S1: Summary of short-answer responses to the post-study questionnaire. Despite the sketch-based web interface being new for everyone, architects performed the task more swiftly on average. It is encouraging that three out of five architects were highly likely to use this sketching interface for massing, though non-architects were less enthusiastic.

S4 User study: Comparison with SketchUp

We tested two more architects, one of whom specializes in urban design. One uses SketchUp regularly; the other routinely works with similar software. Both

were asked to model two scenes, first in our interface and then in SketchUp. In both systems, we provided top-down and perspective references. For SketchUp, we saved the output after 5 minutes and 10 minutes of modeling. After 5 minutes in SketchUp, architects were able to only complete a flat outline of buildings. After 10 minutes, they were still not done with fixing the heights of the buildings, as shown in Fig. S4. Meanwhile, with our system, architects were able to obtain 3D geometry in under 5 minutes. After massing, our models can be exported to detail-oriented modeling tools.

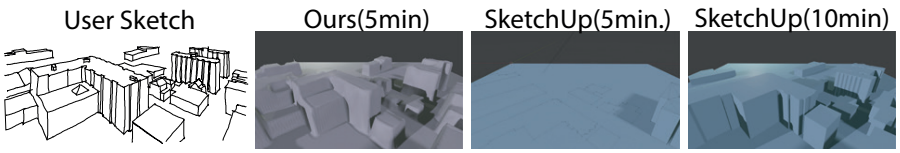


Fig. S4: One of four scenes modeled in GroundUp vs SketchUp by an architect. Qualitatively and quantitatively, quick progress is better in ours.

S5 Perspective depth prediction: Additional analysis and Visualizations

Choice of ν In this section, we analyze the effect of different settings of ν values to construct occupancy feature volume.

The 3D occupancy features are of shape $D \times H \times W$, where D is the number of depth planes. When feeding these features into the 2D encoder in the UNet++, we consider depth planes as image feature channels C. We generate the occupancy features by setting all voxels that fall above non-occupied regions to $-\nu$ and all voxels above occupied regions to ν.

We experiment with 5 different settings: $\nu \in \{1, 25, 50, 75, 100\}$. Tab. S2 shows that using $\nu = 50$ performs better than the other settings. To understand the reason behind this, we observe the range of the multi-scale image features from the image encoder backbone. At the beginning of training, the range of these features is between 0 and 100 for the first few training batches. We believe that keeping ν close to the mid-point of that range allows the network to leverage the occupancy information most beneficially.

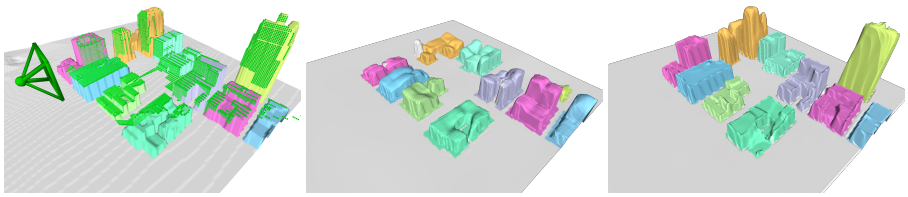
Sparse Height Information Fig. S5 shows height information our diffusion model gets as well as the baseline.

S6 Implementation details

All our models and baselines were trained using PyTorch. For perspective depth prediction, we used a batch size of 16 across all models and ablation experiments,

	Model	Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓	Log RMSE↓	a5↑
1	$OV_{L-\nu 1}$	4.64	3.39	0.31	7.04	5.15	75.1
2	$OV_{L-\nu 25}$	4.8	3.29	0.37	8.41	5.15	74.6
3	$OV_{L-\nu 50}$	**3.49**	**2.13**	**0.21**	**6.54**	**3.43**	**89.2**
4	$OV_{L-\nu 75}$	4.71	3.2	0.33	8.05	4.84	74.5
5	$OV_{L-\nu 100}$	4.22	2.68	0.29	7.74	4.2	81.2

Table S2: The effect of the choice of ν for the Occupancy features Volume (OV) in the perspective depth prediction network. All models are trained using the ResNet-50 encoder. All metric values apart from $a5$ are scaled up by 10^2.



a) b) c)

Fig. S5: Qualitative comparison of *HeightFields* [6] vs our model. a) shows the ground-truth mesh with a camera marker for the perspective view and the visualization of what that view sees overlaid in green. b) is the *HeightFields* [6] output and c) is our model.

with a fixed learning rate of 1e-4 and weight decay. We trained all models for 25K iterations on four RTX 2080 GPUs. Our top-down mask model is trained with similar parameters to the depth predictor except we train it for 5K iterations. For building segmentation, we use Pytorch's BCEWithLogitsLoss function and set *pos_ weight* as 20 for balancing the building pixels against the ground pixels in the mask image. For our depth completion diffusion model, we set a batch size to 32 and a learning rate to $3e-4$. We trained all models for 35 epochs, on a machine with RTX 2080 GPUs. For the depth completion baseline in the main paper, *HeightFields* [6], we used a batch size of 12, a learning rate of $1e-4$, and trained it for 35 epochs on an NVIDIA RTX 3090 GPU. Please note that we added a normal loss $\mathcal{L}_{\text{norm}}$ to this baseline, as we found that results in more accurate reconstructions with sharper features. During training for our models and *HeightFields* [6] baseline, we augment both the top-down and perspective sketches, following the strategy proposed by Ünlü *et al.* [5].

S6.1 Multi-conditional top-down diffusion model

In the main paper, we describe how we condition the diffusion model in Sec. 3.3. Here, we provide additional details of the CNNs that we use to align features of the sketch and depth encoders. The latent features c_{depth} and z_k are passed through separate CNN heads: each head contains two convolutional blocks with a Conv2d layer followed by GroupNorm and Relu layers.

S7 Post-processing heightfields for visualization

In the user interface, we use a quick (real-time) meshing algorithm. We elevate each grid point of an initial 2D flat mesh using the predicted height values, as we described in Sec. 3.3.

To obtain real-time performance, our predicted heightfields have limited spatial resolution, which results in some jagged aliased geometry on the vertical surface of the buildings. This effect can be observed for example in Fig. S5, in both the ground-truth and our predictions, which are both obtained from the same resolution heightfields.

To provide users with an option to work with higher-quality mesh at the next stage of their design pipeline, we explored automatic offline post-processing. We first vectorize the predicted heightfields using Adobe Illustrator's Image Trace tool. We then export it as a high-resolution raster image (300dpi). We use the following settings for Image Trace:

- Preset: custom
- Mode: Grayscale
- Threshold: Between [8-20] (depending on the depth map, the threshold may vary.)
- Paths: 75 / Corners: 75 / Noise: 25

For rendering the vectorized high-resolution output, we used Blender's Render Engine. We used this approach to generate visualizations in the teaser in the main paper (Fig. 1), the supplementary video, and for the visualization of the results of the freehand modeling sessions (Fig. 6 in the main paper).

Potentially, some superresolution approaches that do not require training, such as [2], can also be used to reduce the jaggedness of the reconstructed meshes.

S8 Controllable geometry generation in occluded areas

As sketching is typically the first step in any design process, our primary goal was to enable a tool that combines the benefits of sketching and 3D shape exploration for large-scale city scenes. Depending on the use case, the user interface could be modified to fit different modeling scenarios. Our UI could be extended to allow modeling buildings 1-1 – the strategy chosen during sketching by one of our participants in the user study, *Novice-User-2*. For another scenario, the UI could evolve to support multi-view perspective sketches. Furthermore, one user asked for camera-angle control (see Sec. S3.4).

While we leave a thorough exploration of multi-view iterative editing to future work, we have conducted a preliminary study. To test this, we used 250 test scenes with 2 perspective views 45° apart. We projected point clouds inferred from extra perspective sketches into the top-down representation passed to our diffusion model. Without any finetuning, the reconstruction is improved on all metrics, *e.g.* by .0020 points on absolute difference of top-down view, compared to a single perspective sketch, as seen in Tab. S3.

S9 Effect of normals loss

Model	Abs Diff↓	Abs Rel↓	Sq Rel↓	RMSE↓
GroundUp (single camera)	0.1158	0.0249	0.0073	0.1619
GroundUp (two cameras)	**0.1138**	**0.0244**	**0.0069**	**0.1557**

Table S3: Quantitative evaluation on multi-view input. GroundUp with multi-view input improves metrics.

In Tab. 3 of the main paper, we noticed a drop in performance when the normal loss is used. We tracked this down through visualizations - please see Fig. 4. This loss causes geometry to shrink slightly in all directions - especially in the areas occluded in the perspective sketch. In Fig. 4-a, the red point cloud accounts for both visibility and actual building height. For $\mathcal{L}_{t,norm}$, Fig. 4-b shows the red point cloud is riding slightly above the green prediction, meaning the height is underestimated. In contrast, the prediction without the normal loss does not suffer from underestimated heights within the visibility region, albeit producing uneven surfaces (Fig. 4-c). $\mathcal{L}_{t,norm}$ produces nice building geometry with even surfaces within and outside the visibility region; without it, the model produces unrealistic buildings, deviating a lot from real building geometry, especially outside the visibility region (Fig. 4-c).

The 3D metric in Tab. 3 masks for visibility, so this metric is sensitive to shrinkage while ignoring defects outside the perspective view. The 2D metrics are computed for the full buildings' geometries and reflect on the quality of buildings' rooftops outside of areas visible in the perspective views.

We think the reason for the geometry shrinkage in the visible regions could be explained with the aid of multi-task learning literature. Training a neural network with an auxiliary task could affect the performance on the main task, *e.g.* that of depth and normals estimation [3].

References

1. Blender Online Community: Blender - a 3D modelling and rendering package. Blender Foundation, Blender Institute, Amsterdam (2022), http://www.blender.org
2. Du, R., Chang, D., Hospedales, T., Song, Y.Z., Ma, Z.: Demofusion: Democratising high-resolution image generation with no $$$ (2024)
3. Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., Finn, C.: Efficiently identifying task groupings for multi-task learning. Advances in Neural Information Processing Systems **34** (2021)
4. Lin, L., Liu, Y., Hu, Y., Yan, X., Xie, K., Huang, H.: Capturing, reconstructing, and simulating: The urbanscene3d dataset. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part VIII. Lecture Notes in Computer Science, vol. 13668, pp. 93–109. Springer (2022)

5. Ünlü, G., Sayed, M., Brostow, G.J.: Interactive sketching of mannequin poses. In: International Conference on 3D Vision, 3DV 2022, Prague, Czech Republic, September 12-16, 2022. pp. 700–710. IEEE (2022). https://doi.org/10.1109/3DV57658.2022.00080, https://doi.org/10.1109/3DV57658.2022.00080
6. Watson, J., Vicente, S., Aodha, O.M., Godard, C., Brostow, G.J., Firman, M.: Heightfields for efficient scene reconstruction for AR. In: IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2023, Waikoloa, HI, USA, January 2-7, 2023. pp. 5839–5849. IEEE (2023)